|  |  |  |  |
| --- | --- | --- | --- |
| **Alumno(s):** | LUIS ENRIQUE GUZMAN NIÑO | **Fecha:** | 11/02/2018 |

|  |
| --- |
| OBJETIVO |
| Desarrollar dos aplicaciones para cubrir el tema de los proveedores de contenido, una de ellas tendrá los métodos y la base de datos en SQLite, y la otra modificará los registros de la base de datos de esa aplicación. |

|  |
| --- |
| MARCO TEÓRICO |
| *Un proveedor de contenido administra el acceso a un repositorio central de datos. Un proveedor se implementa como una o más clases en una aplicación de Android, junto con elementos en el archivo de manifiesto. Una de tus clases implementa una subclase* [*ContentProvider*](https://developer.android.com/reference/android/content/ContentProvider.html?hl=es-419)*, que es la interfaz entre tu proveedor y otras aplicaciones. Si bien los proveedores de contenido permiten que los datos estén disponibles para otras aplicaciones, puedes tener actividades en tu aplicación que le permitan al usuario consultar y modificar los datos administrados por tu proveedor.*  *Un proveedor de contenido es la interfaz para los datos guardados en un formato estructurado. Antes de crear la interfaz, debes decidir cómo guardar los datos. Puedes guardar los datos en cualquier formato que quieras y luego diseñar la interfaz para leer y escribir los datos, según sea necesario.*  *Estas son algunas de las tecnologías de almacenamiento de datos disponibles en Android:*  *El sistema Android incluye una SQLite Database API que los proveedores del mismo usan para guardar datos orientados a tablas. La clase*[*SQLiteOpenHelper*](https://developer.android.com/reference/android/database/sqlite/SQLiteOpenHelper.html?hl=es-419)*te ayuda a crear bases de datos y la clase*[*SQLiteDatabase*](https://developer.android.com/reference/android/database/sqlite/SQLiteDatabase.html?hl=es-419)*es la clase de base para acceder a bases de datos.*  *Recuerda que no necesitas usar una base de datos para implementar tu repositorio. Un proveedor aparece externamente como un conjunto de tablas, similar a una base de datos relacional, pero esto no es un requisito para la implementación interna del proveedor.*  *Para guardar datos de archivo, Android tiene una variedad de API orientadas a archivos. Para obtener más información acerca del almacenamiento de archivos, lee el tema*[*Almacenamiento de datos*](https://developer.android.com/guide/topics/data/data-storage.html?hl=es-419)*. Si estás diseñando un proveedor que ofrece datos relacionados con medios, como música o videos, puedes tener un proveedor que combine archivos y datos de tablas.*  *Para trabajar con datos basados en la red, usa clases en*[*java.net*](https://developer.android.com/reference/java/net/package-summary.html?hl=es-419)*y*[*android.net*](https://developer.android.com/reference/android/net/package-summary.html?hl=es-419)*. También puedes sincronizar datos basados en la red con un sitio de almacenamiento local, como una base de datos, y luego ofrecer los datos como tablas o archivos* |

|  |
| --- |
| DESARROLLO |
| *El primer paso que debemos hacer, es terminar la primera aplicación que empezamos en clase, donde teníamos configurada la base de datos del proveedor de contenidos, y los métodos de insertar, eliminar y actualizar declarados, pero no funcionales.*  *Para ello nos dirigimos a la clase ContentProviderContractUser*    *Y en el método insert creamos el siguiente código, donde estamos preparando la base de datos para recibir los datos que se ingresaran en el registro, y retornaremos error o un aviso que confirme si se pudo registrar correctamente el dato en la bd.*    *Para la parte del borrado es algo similar cambiando la instrucción que queremos realizar en la base de datos a “delete”, que nos retornara el registro que vamos a borrar en base al id que le enviemos, por medio de la uri que recibe.*    *Por último, el método actualizar, donde de igual forma cambiaremos la acción a realizar por un “update” y además pondremos la condición que actualizara los datos donde se cumpla la coincidencia del id que deseemos.*    *Ahora pasamos a crear un nuevo proyecto que será el cliente.*  *Creamos un layout sencillo donde pondremos la lista de los datos de la bd y un botón Agregar para abrir un dialogo para ingresar nuevos datos.*  https://scontent.fgdl5-1.fna.fbcdn.net/v/t34.0-12/28053025_1342316895870333_252295919_n.png?oh=8112acf3ac25832f9347affbd86940fa&oe=5A842971  *Y adicionalmente otra interfaz donde estarán los campos del registro que vamos a insertar.*  *Esta ventana la abriremos a modo de dialogo.*    *Creamos la clase UserClientContract*    *En la cual haremos referencia a los campos de la base de datos del proveedor de contenidos usando el siguiente código.*    *En la actividad principal igual haremos referencia al content provider y crearemos arreglos para los datos de los registros de la base de datos.*  *A los elementos de la lista les asignaremos el evento click para obtener los datos de dicho elemento cuando el usuario de click sobre ellos.*    *Al dar click sobre un elemento, se mostrará un cuadro de dialogo con las opciones de eliminar y actualizar el registro seleccionado.*    *Para llenar la tabla con los registros de la bd usamos el siguiente código que se ejecutará reiteradamente y obtendrá el id y nombre, por medio de una conexión del adaptador a nuestra clase UserClientContract.*    *En dicho método tenemos el método getIDs, que es el que recupera los elementos de la tabla. Y los añade uno a uno a las listas que declaramos al inicio de este programa.*    *Creamos otro método, donde abriremos un dialogo, con la interfaz de nuestro segundo layout donde pedimos los datos, y dicho layout, tendrá datos para actualizar o estará vacío dependiendo de la forma en la que el usuario acceda a dicho formulario.*    *Al momento de dar click en el botón guardar de dicho dialogo, dependiendo de la opción con la que haya entrado el usuario a ese dialogo, se ejecutara una de las dos acciones, ya sea insertando registro, tomando todos los datos y añadiéndolos a un nuevo registro con un id autoincrementadle, o actualizando los datos ya existentes, con los que pongamos dentro de ls campos de texto en un id de un elemento de la tabla ya existente.*  *Si el usuario en el cuadro de dialogo selecciona la opción de eliminar se ejecutará el siguiente código, donde eliminamos los datos de la lista, y además llamamos el método delete del content resolver enviando el id del usuario que queremos eliminar a la uri del content resolver y actualizando la lista.* |

|  |
| --- |
| CONCLUSIONES |
| *Ya que las URIs definidas por un Proveedor de contenidos (CV) son únicas para ese proveedor, es muy importante que estas URIs estén bien documentadas. Los proveedores que Android proporciona hacen esto definiendo constantes que representan las cadenas de la URI.*  *El proveedor*MediaStore*define dos URIs y*Contacts*uno. Si te das cuenta, las constantes están definidas usando esquema jerárquico. Por ejemplo, la URI de los contactos se señala como*Contacts.People.CONTENT\_URI.*Esto se debe a que las bases de datos de contactos pueden tener muchas tablas para representar la entidad de un contacto.*People*es una de las tablas o colecciones. Cada entidad primaria de una base de datos puede llevar su propia URI de contenido.* |

|  |
| --- |
| BIBLIOGRAFÍA |
| [*https://developer.android.com/guide/topics/providers/content-providers.html?hl=es-419*](https://developer.android.com/guide/topics/providers/content-providers.html?hl=es-419) |